Granular clustering of *de novo* protein models

Dmytro Guzenko[1] and Sergei V. Strelkov*[1]

[1]Department of Pharmaceutical and Pharmacological Sciences, KU Leuven, Leuven 3000 , Belgium.

1 Abstract

Motivation: Modern algorithms for *de novo* prediction of protein structures typically output multiple full-length models (decoys) rather than a single solution. Subsequent clustering of such decoys is used both to gauge the success of the modelling and to decide on the most native-like conformation. At the same time, partial protein models are sufficient for some applications such as crystallographic phasing by molecular replacement (MR) in particular, provided these models represent a certain part of the target structure with reasonable accuracy.
Results: Here we propose a novel clustering algorithm that natively operates in the space of partial models through an approach known as granular clustering (GC). The algorithm is based on growing local similarities found in a pool of initial decoys. We demonstrate that the resulting clusters of partial models provide a substantially more accurate structural detail on the target protein than those obtained upon a global alignment of decoys. As the result, the partial models output by our GC algorithm are also much more effective towards the MR procedure, compared to the models produced by existing software.
Availability: The source code is freely available at https://github.com/biocryst/gc
Contact: sergei.strelkov@kuleuven.be
Supplementary information: Supplementary data are available at *Bioinformatics* online.

2 Introduction

Protein *de novo* 3-dimensional (3D) structure prediction involves extensive sampling of the conformation space in search of the near-native low energy state. The large number of decoys produced makes it impossible to inspect and interpret the results manually. Structural clustering is a widely used tool for post-processing of *de-novo* folded decoys [8], [22]. It exploits the idea that frequently sampled low-energy conformations are more likely to represent the native structure than the lone lowest-energy decoy [16].

Clustering algorithms require a dissimilarity measure between any two objects. This function involves a superposition of the structures that optimises certain score, most typically a root-mean-square deviation (RMSD) of atomic positions minimised with the Kabsch algorithm [7]. Clearly, a single superposition of full-length models often fails to reveal a complete information on their local similarities. An obvious model situation is a protein consisting of two domains connected by a flexible linker. Such protein can accept a multitude of conformations that are globally very different, even though the conformations of individual domains remain the same. Assessment of

*To whom correspondence should be addressed.

local model quality independently of domain motions has long been implemented during the Critical Assessment of Methods for Structure Prediction of Proteins (CASP) competition, with specialised metrics continuously developed [20], [10]. However, clustering algorithms routinely used to post-process decoys generated by *de-novo* protein folding are still based on single-alignment approaches.

Lack of sensitivity for local similarities inherently limits the capabilities of the cluster analysis towards extracting useful information from the pool of decoys. To overcome this difficulty, one can generate more decoys, hoping that the correct global fold reveals itself as a statistically significant cluster. This is a viable approach if the aim is to obtain an accurate full-length model, but it requires significant computational resources and specific optimisations to handle large distance matrices [23], since hundreds of thousands of decoys are not uncommon.

Here we describe a new method to obtain partial protein models which is based on the granular clustering (GC) paradigm [13]. It outputs substructures that are similar in a sufficiently large number of decoys, without prior assumptions on the modelling accuracy or the substructure size. Clusters of full-length models are the ultimate possibility, making this method complementary to the clustering based on global alignment, *i.e.* working bottom-up, *vs.* top-down, towards the same goal.

Our method is especially useful in applications where partial models covering different fragments of the protein sequence, possibly providing alternative conformations, are sufficient. An important example of such an application is solving the "phase problem" in X-ray crystallography by molecular replacement (MR). Traditionally, the MR procedure required the availability of an experimentally determined protein structure that is sufficiently homologous to the target protein [15]. More recently, as the methods for *ab initio* structure prediction continued to improve, the possibility to use predicted (partial) structures in MR searches has been demonstrated [4], [21], [1]. Here we show that our partial models obtained through GC are twice as effective in the MR procedure, compared to models prepared using existing approaches from the same initial pool of decoys. Further applications of GC of protein models may include local contacts prediction, non-linear structural motifs discovery, or generation of custom libraries of structural fragments [14].

3 System and methods

3.1 Principles of granular clustering

We formulate the problem within the granular computing paradigm [19], which is particularly suited for our bottom-up approach towards obtaining clusters of partial models. The general principles of GC [13] are as follows:

- Primitive *information granules* are created from the input data elements; these are subsets of the data that can be directly aggregated by a specific property.
- Clustering is carried out by *growing* information granules – iteratively merging granules that have significant overlap;
- Clustering is stopped when enough data *condensation* is achieved.

The criteria for information granulation, granule merging and data condensation are not generally defined and are specific for a particular application.

3.2 Granular clustering of protein decoys

We start by defining the granular protein clustering as a search problem, *i.e.*, in terms of initial state, production rule and the goal state. Organisation of the search itself will be discussed in the next section.

Let us consider a set of K decoys for a given protein containing N amino acid residues. Then $\mathbf{R} = \{1, ..., N\}$ is a set of numbered residues of this protein and $\mathbf{M} = \{1, ..., K\}$ is a set of numbered decoys. Cluster C, which may contain only some stretches along the sequence rather than the complete protein, and represented by only some of the available decoys, is defined as a pair $C = (R, M), R \subseteq \mathbf{R}, M \subseteq \mathbf{M}$.

Let $\mathrm{V}(R, M)$ be a scoring function of a cluster, subject to minimisation. An obvious example of such function is $\mathrm{RMSD}(R, M)$ – the average root-mean-square deviation between all pairs of decoys $m_i, m_j \in M, i \neq j$ evaluated on the optimal superposition of Cα atoms of residues R.

A cluster is *valid* with respect to parameters (v, s) if it consists of s or more models, superposition of which by the given residues R produces score of at most v.

$$\mathrm{Valid}(R, M|v, s) \iff \begin{array}{l} |M| \geq s \\ \mathrm{V}(R, M) \leq v \end{array} \tag{1}$$

A cluster is *saturated* with respect to parameters (v, s), if no further models can be added to the cluster without breaking its validity.

$$\mathrm{Saturated}(R, M|v, s) \iff \begin{array}{l} \mathrm{Valid}(R, M|v, s) \\ \nexists M' \supset M : \mathrm{Valid}(R, M'|v, s) \end{array} \tag{2}$$

Let $\mathbf{C}(R, M|v, s)$ be a set of all saturated clusters with respect to parameters v, s, further denoted as $\mathbf{C}$ for brevity.

Initial state. The set of all clusters $\mathbf{C}$ contains all one-residue segments with the entire set of models as support, i.e.

$$(\{i : i \in \mathbf{R}\}, \mathbf{M}) \in \mathbf{C} \tag{3}$$

Since each of these clusters contain all models to begin with, no further models can be added. Thus the saturation condition is naturally satisfied.

Production rule. Two distinct clusters can be combined (*i.e.* condensed) if they share enough supporting models and stay below a given score limit. In order to ensure eventual algorithm termination, we require that each input cluster for the production rule contains at least one residue not found in another cluster. Let $C_1 = (R_1, M_1) \in \mathbf{C}$, $C_2 = (R_2, M_2) \in \mathbf{C}$, where R_1 and R_2 are not subsets of each other, then a set of clusters produced by C_1 and C_2 is defined as:

$$\mathrm{Prod}(C_1, C_2|v, s) = \{(R, M) \in \mathbf{C} : R = R_1 \cup R_2, M \subseteq M_1 \cap M_2\} \tag{4}$$

Note that there may be several clusters produced by one pair of inputs. This may happen when two substructures jointly adopt alternative conformations that are sufficiently supported by the pool of decoys. For example, parallel and antiparallel configurations of α-helical chains are drastically different structural arrangements, yet they may be regulated by subtle changes in hydrophobic core packing energy [9].

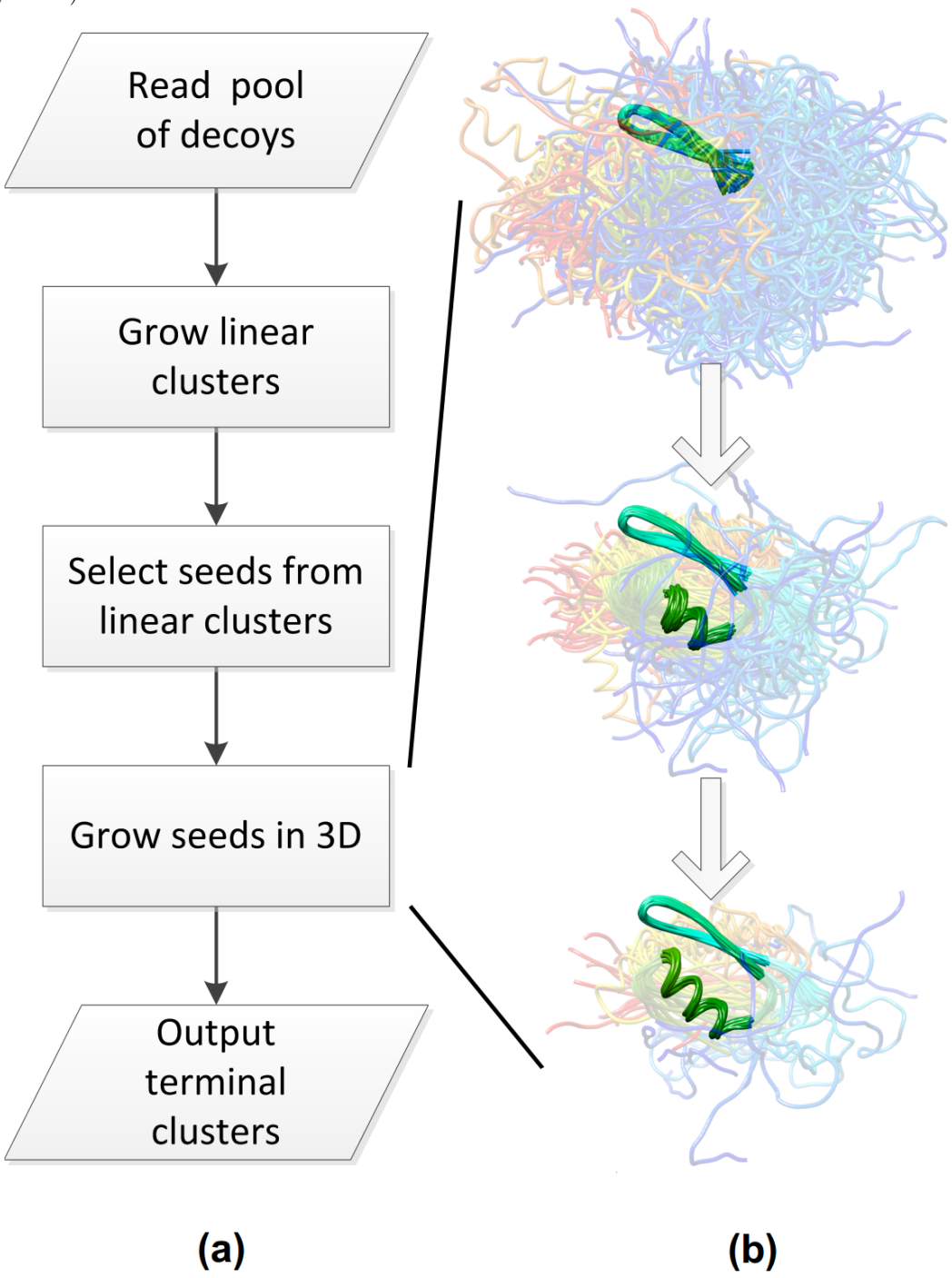
A cluster is *terminal* if no new clusters can be produced using it as one of the inputs.

$$Prod(C_T, C|v, s) = \varnothing, \forall C \in \mathbf{C} \tag{5}$$

The *goal state* of the granular clustering is then defined by finding all terminal clusters.

The clustering problem is now formulated as a classical combinatorial search, which enables application of the wealth of methods developed for this purpose. As a proof of concept, we implemented a search by greedy heuristic with a number of simplifications, which allows reaching suboptimal yet evidently useful results within a short computational time.

Figure 1: (**a**) Flowchart of the heuristic GC implementation. (**b**) An example of 3D cluster growing step. The supporting decoys for each step are shown as transparent ribbon diagrams. All structures are coloured by a gradient from blue (N-terminus) to red (C-terminus). Starting from a 16-residue seed the cluster is grown by 8 residues, which are not linear in sequence, then extended by 4 more residues, after which no more suitable candidates found and the procedure is terminated. The illustration is based on the 500 Rosetta decoys produced for a 111 residues long protein target (PDB entry 2C60).



4 Algorithm

4.1 Overview

The naive exhaustive search of all terminal clusters would involve costly all-*vs*-all RMSD minimisations, the number of which grows quadratically with the number of models. Moreover, all possible subsets of all residues have to be considered, the number of which grows exponentially with the sequence length. To tackle the computational complexity we split the problem into a two-step procedure.

A short stretch of consecutive residues with similar backbone torsion angles is likely to have a low structural variability in the pool of decoys. We can exploit this fact to quickly grow granules linearly in protein sequence by iteratively merging the initial one-residue clusters (3). The results of this step serve as input for the full-scale 3D granular clustering procedure. Hence it can be viewed as preprocessing to reduce the search space. The overall procedure is illustrated in Fig. 1a.

The implementation of the production rule (4) is split into two parts. Initially, the two input clusters are combined into one, using a simplified production rule:

$$\mathrm{Prod}'(C_1, C_2) = (R_1 \cup R_2, M_1 \cap M_2) \tag{6}$$

Afterwards, the problem of alternative conformations and output cluster validity is solved by *subclustering* the result (6). Specifically, let $D(R, M)$ be a matrix $|M|\times|M|$ of distances between the models in M evaluated on residues in R. Let $\mathcal{C}(D|h)$ be a standard clustering algorithm parametrised by some constant h, such as the number of clusters or a density threshold. The algorithm takes distance matrix D as an input and produces clusters $\{M_1, ..M_K\}$. We define subclustering procedure $\mathrm{Sub}(C|D)$ (parameters v, s, h omitted for brevity) of a cluster $C = (R, M)$ using distance information $D(R, M)$ as follows:

$$\begin{gathered}\mathrm{Sub}(C|D, v, s, h) = \bigcup_{M_i \in \mathcal{C}(D|h)} \{(R, M_i) : \mathrm{Valid}(R, M_i|v, s)\} \\ \bigcup_i M_i \subseteq M; M_i \cap M_j = \varnothing, i < j \leq K\end{gathered} \tag{7}$$

In this implementation, the cluster validity (1) is ensured, while the requirement for cluster saturation (2) is relaxed and depends on the properties and parameters of the standard clustering algorithm chosen. Note that the definition (7) implies deterministic rather than probabilistic cluster assignments, but can be easily generalised for the latter.

4.2 Linear cluster growing

Let $\mathbf{S}^0 = \bigcup_{1<k<N} \left\{(\{k\}, \mathbf{M})\right\}$ be the initial state of one-residue clusters. Each cluster has an associated distance matrix in the space of backbone torsion angles $D_{\phi,\psi}(\{k\}, \mathbf{M})$. The first and the last residues are omitted, since they will lack either ϕ or ψ by definition.

The successive states $\mathbf{S}^{i+1}$ are constructed by application of the production rule (6) with subsequent subclustering (7) for every two clusters from $\mathbf{S}^i$ that describe adjacent segments by the residue number. Let $C_1 = (R_1, M_1) \in \mathbf{S}^i$, $C_2 = (R_2, M_2) \in \mathbf{S}^i$. Then their product is

$$\mathbf{P}^{i+1} = \bigcup_{C_1, C_2 \in \mathbf{S}^i} \{\mathrm{Prod'}(C_1, C_2) : \max R_1 = \min R_2 - 1\} \tag{8}$$

And the next state is given by

$$\mathbf{S}^{i+1} = \bigcup_{C \in \mathbf{P}^{i+1}} \mathrm{Sub}(C|D_{\phi,\psi}) \tag{9}$$

By definition (9), $\mathbf{S}^i$ contains clusters of length 2^i, The procedure will eventually terminate on its own, when either no valid clusters can be produced or 2^i surpasses the sequence length. However, structural variability between the supporting models will become less predictable with the cluster length, even if the backbone torsion angles are similar, so imposing a reasonable limit of i_{max} is necessary. Thus the output of the first level of granular clustering is a set of terminal clusters $\mathbf{S}^{i_{max}}$ and a set of intermediate clusters $\mathbf{L} = \bigcup_{0<i<i_{max}} \{\mathbf{S}^i\}$.

4.3 Seed selection

Next, we need to select the "seeds" (*i.e.* the initial state) of 3D cluster growing. It would be reasonable to start from the longest segments available, $\mathbf{S}^{i_{max}}$. However, they may overlap by residue numbers and supporting models to a large extent. We have observed that such overlapping clusters, if grown, frequently produce similar end-results (data not shown), and eliminating them would only slightly diminish the total coverage, while greatly reducing the search space. To this end, a non-redundant set of seeds from the longest linear clusters $\mathbf{S}_{seeds} \subseteq \mathbf{S}^{i_{max}}$ is constructed as follows.

Here we use the Jaccard index to quantify similarity of two sets $\mathrm{Jac}(A, B) = \frac{|A \cap B|}{|A \cup B|}$. Two clusters $C_1 = (R_1, M_1)$, $C_2 = (R_2, M_2)$ are considered independent with respect to parameter J_{max} if the sets of their residues and supporting models have pairwise similarity of at most J_{max}:

$$\mathrm{Ind}(C_1, C_2 | J_{max}) \iff \begin{array}{l} \mathrm{Jac}(R_1, R_2) \le J_{max} \\ \mathrm{Jac}(M_1, M_2) \le J_{max} \end{array} \tag{10}$$

A set of clusters $\mathbf{S} = \{(R_1, M_1), ..., (R_k, M_k)\}$ is considered independent with respect to J_{max} if all possible pairs of clusters in the set are independent with respect to J_{max}:

$$\mathrm{Ind}(\mathbf{S} | J_{max}) \iff \operatorname*{Ind}_{\forall C_i, C_j \in \mathbf{S}, i \neq j} (C_i, C_j | J_{max}) \tag{11}$$

Let $\mathbf{S}^{i_{max}}_{sorted} = \Big\langle C_1 = (R_1, M_1), ..., C_k = (R_k, M_k) \Big\rangle$ be a sequence of clusters from $\mathbf{S}^{i_{max}}$ ordered by decreasing support, i.e. $|M_{j-1}| \geq |M_j|, j = 2..k$. A set of seeds $\mathbf{S}_{seeds}$ is defined as follows:

1. The cluster with the largest support is included into the seeds set: $C_1 \in \mathbf{S}_{seeds}$.
2. Each subsequent cluster from $\mathbf{S}^{i_{max}}_{sorted}$ is included into the seeds set if it does not break the independence criterion with the already included clusters:

$$C_i \in \mathbf{S}_{seeds} \iff \mathrm{Ind}\Big(\bigcup_{j<i} \{C_j : C_j \in \mathbf{S}_{seeds}\} \cup C_i \Big) \tag{12}$$

4.4 3D cluster growing

At the start we have the set of seeds $\mathbf{S}_{seeds}$ and the set of linear clusters $\mathbf{L}$, grouped by their lengths 2 to $2^{i_{max}-1}$. Here we search for clusters that are similar in 3D (but may be non-linear in sequence), using the distance matrix $D_{rmsd}(R, M)$ defined by pairwise RMSD of atomic coordinates between all models in M on residues in R.

The high-level algorithm for growing a cluster from a seed is presented in Algorithm 1. The procedure $\mathrm{Select}(C, \mathbf{S})$ is a greedy heuristic for cluster extension and involves the following computations. First, we produce all possible candidate extensions of a seed $C \in \mathbf{S}_{seeds}$ with the given set of clusters $\mathbf{S} \in \mathbf{L}$ using production rule (6):

$$\mathrm{Combine}(C, \mathbf{S}) = \bigcup_{S \in \mathbf{S}} \mathrm{Prod}'(C, S) \tag{13}$$

Then the candidate extensions are subclustered (7) and the results are aggregated into one set:

$$\mathrm{Split}(C, \mathbf{S}) = \bigcup_{C' \in \mathrm{Combine}(C, \mathbf{S})} \mathrm{Sub}(C' | D_{rmsd}(C')) \tag{14}$$

Finally, the best-scoring subcluster is selected:

$$\mathrm{Select}(C, \mathbf{S}) = \operatorname*{argmin}_{C' \in \mathrm{Split}(C, \mathbf{S})} V(C'), \tag{15}$$

where $V(C)$ is a cluster scoring function. In case the folding algorithm provides scores for individual decoys, such as Rosetta energy function [14], they may be included in the evaluation of the clusters. The scoring functions are generally designed to provide prediction of the likeliness to the native structure and could therefore be helpful towards estimating the quality of a resulting cluster. We define $V(C)$ as sum of Rosetta energy scores E ($E = -1$ if energies are not available) of the decoys in a cluster C divided by the average pairwise superposition RMSD $\mathcal{R}(C)$ of the residues included in the cluster.

$$V(C) = \frac{\sum_{i \in M} E_i}{\mathcal{R}(C) + 1} \tag{16}$$

Algorithm 1 Cluster growing

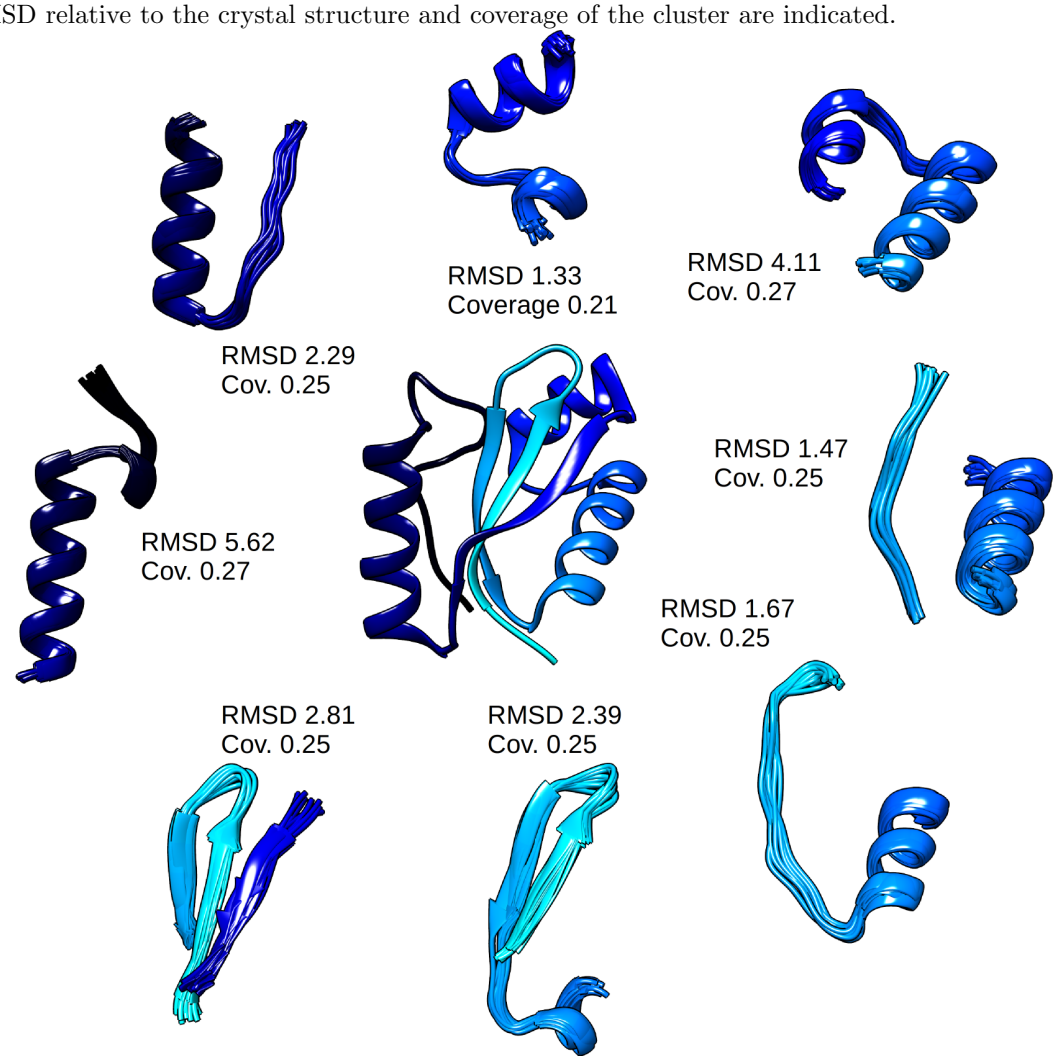
Require: $C_{seed} \in \mathbf{S}_{seeds}, \mathbf{L} = \{\mathbf{S}^1, ..., \mathbf{S}^{i_{max}-1}\}$;
procedure GROWCLUSTER($C_{seed}, \mathbf{L}$)
  $C_{cur} := C_{seed}$
  $i_{cur} \leftarrow i_{max} - 1$
  repeat
    $C_{cand} \leftarrow \mathrm{Select}(C_{cur}, \mathbf{S}^{i_{cur}})$ ▷ Select the best extension
    if $C_{cand} = \varnothing$ **then**
      $i_{cur} \leftarrow i_{cur} - 1$
    else
      $C_{cur} \leftarrow C_{cand}$
    end if
  until $i_{cur} = 0$ ▷ No more candidate clusters
  return C_{cur}
end procedure

5 Implementation

The GC algorithm is implemented as a Python script. Biopython [2] is used to process the Protein Data Bank (PDB) files. Structure superpositions and RMSD calculations are done with PyRMSD [6]. Subclustering is performed with the mean-shift (MS) algorithm [3]. Backbone torsion angles are used directly as samples for MS, while all-*vs*-all RMSD matrices are firstly embedded into a 2D space with multidimensional scaling (MDS). Standard MS and MDS implementations from Scikit-learn [12] package are used.

The algorithm uses a number of parameters that affect the end-result in various ways (Table 1). The default values indicated were found to be a reasonable starting point for problem-specific fine-tuning.

Figure 2: Ribbon diagrams of a 97 residues long protein target in the centre (PDB entry 1MK0) and sample clusters obtained by GC containing 10 models each. All structures are coloured by a gradient from light blue (N-terminus) to dark blue (C-terminus). For each cluster, the average Cα RMSD relative to the crystal structure and coverage of the cluster are indicated.



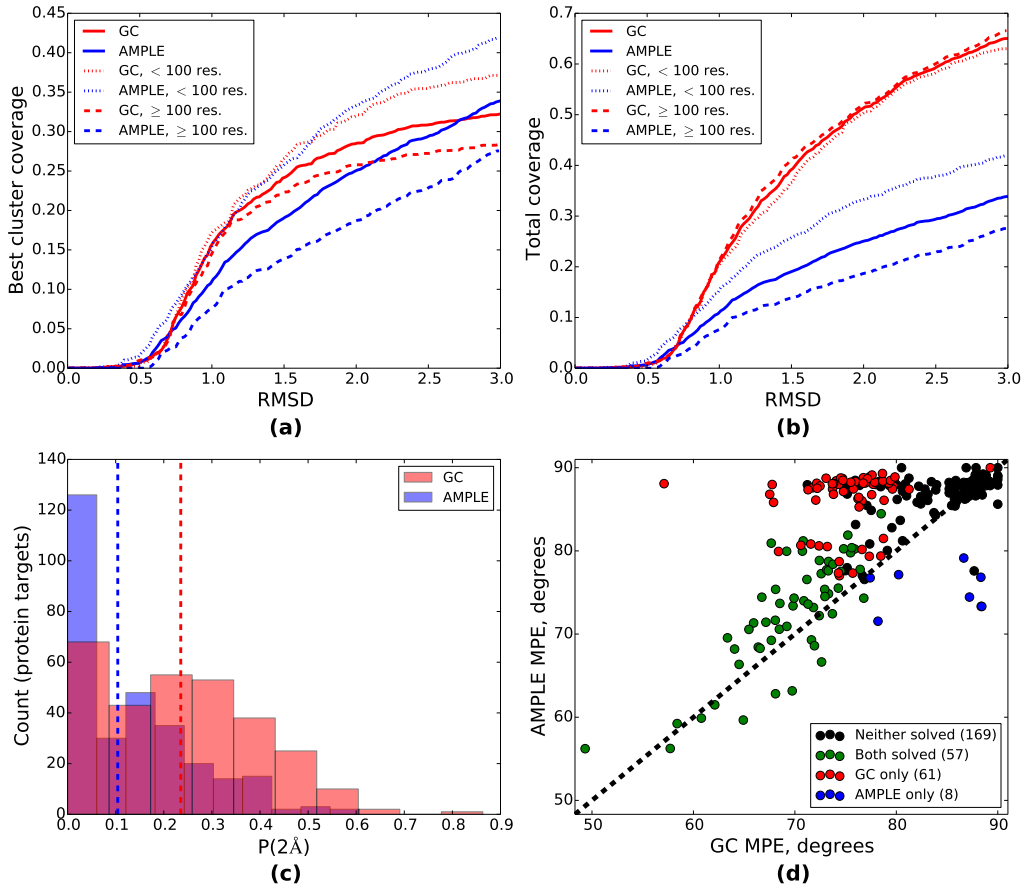


Figure 3: Evaluation of the GC algorithm. (**a**) Best cluster coverage as a function of the RMSD value to the true structure (18). The values averaged over the benchmark set of 295 target structures are plotted. `AMPLE` results plotted in blue, GC – in red. Solid line gives mean values for the entire dataset, dotted line represents proteins with sequence length less than a 100 residues, dashed line – proteins 100 residues and longer. (**b**) The corresponding total coverage function (19). (**c**) Histogram of integral coverage (20), calculated for all clusters produced by `AMPLE` (blue) and GC (red) that are within 2Å RMSD to the true structure. The median values are shown by vertical dashed lines: 0.1 for `AMPLE`, 0.24 for GC. (**d**) Scatter plot comparing the results of the MR search using clusters obtained through both algorithms. Each point corresponds to a target from the benchmark set. The x and y axes give the minimal MPE values among the MR solutions for this target obtained with GC clusters and `AMPLE` clusters respectively. Points above the diagonal represent structures where GC clusters yielded MR solutions closer to the true structure than the `AMPLE` clusters, while the opposite is true for the points below the diagonal. Colour-coding indicates the outcome of automatic model rebuilding in `SHELXE` of the respective MR solutions: failed with both `AMPLE` and GC clusters (black), succeeded with both `AMPLE` and GC clusters (green), succeeded with GC, but not with `AMPLE` (red), succeeded with `AMPLE`, but not GC (blue).

Table 1: GC algorithm parameters.

Parameter	Description	Affects	Default
h_{lin}	Mean-shift bandwidth for subclustering in the space of torsion angles	Precision of linear clusters	1.2
s_{lin}	Minimal support for the linear cluster growing	Number of linear clusters	10% of the pool
i_{max}	Number of repeats of the cluster doubling procedure	Maximal length of seeds	4
J_{max}	Maximal overlap between seed candidates	Number of seeds	0.5
h_{rmsd}	Mean-shift bandwidth for subclustering in the space of RMSD distances	Precision of output clusters	0.5
s_{rmsd}	Minimal support for the 3D cluster growing	Length of output clusters	10

6 Results

A test dataset of 295 crystal structures [1] was used to evaluate the performance of the GC algorithm. 500 decoys for each target were folded using Rosetta [8]. We compared the partial protein clusters produced by the GC algorithm using default parameters to the clusters generated by the `AMPLE` pipeline [1]. This pipeline contains a cluster-and-truncate component which is the most comparable method result-wise, while conceptually being the opposite, since the clusters are generated by global alignment and elimination of diverging segments, with subsequent re-clustering. Additionally, we have compared the relative success of the MR search models provided by the two methods.

6.1 Coverage

To evaluate the clusters quality *per se*, we estimated their *coverage* (in terms of fraction of the total sequence length) as a function of the maximal allowed RMSD from the true structure. Let average RMSD of models in the cluster (R, M) with respect to the true experimental structure $\mathbf{N}$ be given by $\mathcal{R}(R, M|\mathbf{N})$. For a set of output clusters $\mathbf{C}^O = \{(R_1, M_1), ..., (R_{N_O}, M_{N_O})\}$ a subset of clusters that are within an RMSD of r from the true structure is given by

$$\widetilde{\mathbf{C}}^O(\mathbf{N}|r) = \{(R_i, M_i) \in \mathbf{C}^O, \mathcal{R}(R_i, M_i|\mathbf{N}) \leq r\} \tag{17}$$

We will consider the coverage of a single 'best' cluster (understood as the cluster containing the largest number of residues) from this subset

$$\mathrm{Cov}_s(\mathbf{C}^O, \mathbf{N}|r) = \frac{\max_i\{|R_i \cap \mathbf{N}|\colon (R_i, M_i) \in \widetilde{\mathbf{C}}^O(\mathbf{N}|r)\}}{|\mathbf{N}|} \tag{18}$$

We also define the total coverage of this subset as

$$\mathrm{Cov}_t(\mathbf{C}^O, \mathbf{N}|r) = \frac{\left|\bigcup_i \{R_i \cap \mathbf{N} : (R_i, M_i) \in \widetilde{\mathbf{C}}^O(\mathbf{N}|r)\}\right|}{|\mathbf{N}|} \tag{19}$$

A set of clusters output by the GC algorithm for a protein target, their RMSD to the true structure and coverage are presented in Fig. 2. As can be seen, all regions are covered with variable precision, sometimes offering different conformations for the same segment. It is worth mentioning that `AMPLE` clusters (Supplementary Fig. S1, S2) follow a completely different pattern: starting from a small core of structurally conserved residues they gradually increase in coverage while becoming more divergent.

Averaging of the $\mathrm{Cov}_s(r)$ and $\mathrm{Cov}_t(r)$ functions over the benchmark set of 295 target structures can give an idea about the comparative performance of the two methods (Fig. 3a,b). In terms of the best coverage by a single cluster, GC procedure clearly outperforms the `AMPLE` routine for all reasonably precise clusters (up to 2.5Å RMSD from the target structure), with the advantage even more pronounced for longer sequences. In addition, the collective coverage by all clusters output by the GC algorithm is also much better, exceeding that of `AMPLE` by a factor of two for almost any RMSD cut-off (Fig. 3b). The sequence length has a negligible effect on the total coverage (dotted and dashed lines on Fig. 3b). This implies that longer protein chain modelled by Rosetta can still produce useful results if local clusters are our target.

In addition, to assess the distribution of cluster quality for individual targets, we have analysed the integral coverage function $P(\mathbf{C}^O, \mathbf{N}|r_{max})$, which is analogous to the area under curve for a receiver operating characteristic [5]. The maximal acceptable level of RMSD deviation r_{max} is used to bring the metric to the scale of 0 to 1. Let $\langle r_1, ..., r_{N_O} \rangle$ be a sequence of cluster RMSDs to the native structure $\mathcal{R}(R_i, M_i|\mathbf{N})$ sorted from smallest to largest, i.e. $r_i \leq r_{i+1}$. This integral coverage function is calculated using a trapezoidal formula:

$$P(\mathbf{C}^O, \mathbf{N}|r_{max}) = \frac{1}{2r_{max}} \sum_{k=1}^{N-1} (r_{k+1} - r_k) \left(\mathrm{Cov}_t(\mathbf{C}^O, \mathbf{N}|r_{k+1}) + \mathrm{Cov}_t(\mathbf{C}^O, \mathbf{N}|r_k)\right), \tag{20}$$

where N is the number of clusters with $r_k \leq r_{max}$, and $\mathrm{Cov}_t(r_k)$ – total coverage at r_k as defined by (19).

Fig. 3c shows comparison of P distributions calculated on the test set for $r_{max} = 2$Å (see also Supplementary Fig. S3). Here again the superior capabilities of the GC algorithm are very apparent, with more than two-fold increase in median integral coverage compared with the `AMPLE` output. Overall the statistics presented indicate the strength of the local bottom-up approach used in GC.

6.2 Use for molecular replacement

We have also evaluated the usefulness of GC-based partial models towards phasing crystal structures of proteins by MR on the same test set of 295 crystal structures of non-homologous proteins [1]. Previously these authors have explored the use of `AMPLE`-derived clusters towards phasing this test set by first running the MR procedure and thereafter attempting structure rebuilding and extension of this solution using `SHELXE` [17], repeating the whole calculation for every cluster. Here we have employed a further modification of this routine, which allows testing of the clusters' performance in a radically reduced computational time. Initially, for each target structure an MR search using the obtained clusters as search models was performed by `Phaser` [11]. At this point, all MR solutions obtained with various clusters for a given target were evaluated with respect to the similarity to the true crystal structure. To this end, calculation of their mean phase error (MPE) with respect to the true structure was performed using `cphasematch` [18]. Thereafter, only the MR solution yielding

the lowest MPE underwent structure rebuilding. A case was considered solved if this procedure could advance beyond a certain minimal chain length and correlation between the rebuilt model and electron density, as evaluated in `SHELXE`. Further details as well as a table with complete results are provided in the Supplement. Fig. 3d summarises the performance of GC and `AMPLE` clusters as MR search models. In more than two thirds of the test cases lower MPE values could be achieved with the GC clusters. The value of 80° appears to be a cut-off beyond which the automatic model extension is unlikely to succeed, typically because the MR solution has been completely wrong in the first place. In 144 cases, the minimal MPE of the MR solutions obtained with GC clusters was below the said cut-off, compared to 70 such cases with the `AMPLE` clusters. A majority of these solutions could be successfully rebuilt and expanded in `SHELXE`. Ultimately, only 65 of the 295 test cases could be successfully phased using `AMPLE` clusters, while 118 structures could be phased with GC models. The use of GC-based models has thus resulted in an about two-fold higher success rate of the MR procedure.

Of further note, the granular approach to search model generation frequently results in clusters that do not overlap by sequence. This means that during the MR procedure one can attempt to place two or more independent search models at once. This approach has allowed us to obtain a correct MR solution in at least one additional case (1SBX, data not shown).

7 Discussion

Here we have proposed a novel method to produce clusters of partial models from protein decoys based on local structural similarity, which falls under the granular computing paradigm. It should be noted that the existing methods of protein model clustering typically consider a full-length decoy as a single data point; all such decoys are then clustered upon some sort of a single-alignment procedure. This imposes an obvious limitation on the clustering algorithm, since one can not operate with less than a whole decoy. In contrast, the GC approach operates in a much larger search space, since the decoy data are initially granulated down to the level of a single residue. As we have shown here, this enables the design of a clustering algorithm that is very efficient in extracting the structural information from a pool of *de-novo* modelled decoys. While more demanding computationally compared to approaches based on a single structural alignment, GC is nevertheless capable of yielding extremely useful results for typical proteins even when using modest computational resources.

Our implementation is the first 'proof-of-concept' of the GC approach to protein structures, and application of more advanced heuristic search strategies is likely to follow. Moreover, we envisage further development of this approach towards a range of research questions. In particular, this could include algorithms to detect non-linear structural motifs in a large set of 3D structures, such as the experimental structures available in the PDB. In addition, by incorporating amino acid sequence distribution in the observed clusters, one could obtain variable-length fragment library for protein structure prediction. In this case, fragments with long-distance interactions could be used for the generation of prior spatial constraints to be utilized during the *ab initio* protein modelling.

In conclusion, we have developed an alternative view of the structural protein clustering problem, which enables 'growing' clusters of partial models from local similarities observed in sampled conformations. We have shown that solving the phase problem in X-ray crystallography is an area that can immediately benefit from the results obtained here. We hope they will facilitate development of further novel techniques in protein structure prediction as well as aid in experimental structure determination.

References

[1] Jaclyn Bibby, Ronan M. Keegan, Olga Mayans, Martyn D. Winn, and Daniel J. Rigden. AMPLE: a cluster-and-truncate approach to solve the crystal structures of small proteins using

rapidly computed ab initio models. *Acta Crystallographica Section D*, 68(12):1622–1631, Dec 2012.

[2] Peter J. A. Cock, Tiago Antao, Jeffrey T. Chang, Brad A. Chapman, Cymon J. Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, and Michiel J. L. de Hoon. Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423, 2009.

[3] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

[4] Rhiju Das and David Baker. Macromolecular modeling with Rosetta. *Annu. Rev. Biochem.*, 77:363–382, 2008.

[5] Tom Fawcett. An introduction to ROC analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[6] Vctor A. Gil and Vctor Guallar. pyRMSD: a Python package for efficient pairwise RMSD matrix calculation and handling. *Bioinformatics*, 29(18):2363–2364, 2013.

[7] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.

[8] Andrew Leaver-Fay, Michael Tyka, Steven M Lewis, Oliver F Lange, James Thompson, Ron Jacak, Kristian Kaufman, P Douglas Renfrew, Colin A Smith, Will Sheffler, et al. ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. *Methods in enzymology*, 487:545, 2011.

[9] Vladimir N Malashkevich, Chelsea D Higgins, Steven C Almo, and Jonathan R Lai. A switch from parallel to antiparallel strand orientation in a coiled-coil X-ray structure via two core hydrophobic mutations. *Peptide Science*, 104(3):178–185, 2015.

[10] Valerio Mariani, Marco Biasini, Alessandro Barbato, and Torsten Schwede. lDDT: a local superposition-free score for comparing protein structures and models using distance difference tests. *Bioinformatics*, 29(21):2722–2728, 2013.

[11] Airlie J. McCoy, Ralf W. Grosse-Kunstleve, Paul D. Adams, Martyn D. Winn, Laurent C. Storoni, and Randy J. Read. Phaser crystallographic software. *Journal of Applied Crystallography*, 40(4):658–674, Aug 2007.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[13] W. Pedrycz and Andrzej Bargiela. Granular clustering: a granular signature of data. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 32(2):212–224, Apr 2002.

[14] Carol A Rohl, Charlie EM Strauss, Kira MS Misura, and David Baker. Protein structure prediction using Rosetta. *Methods in enzymology*, 383:66–93, 2004.

[15] MICHAEL G Rossmann. The molecular replacement method. *Acta Crystallographica Section A: Foundations of Crystallography*, 46(2):73–82, 1990.

[16] David Shortle, Kim T. Simons, and David Baker. Clustering of low-energy conformations near the native structures of small proteins. *Proceedings of the National Academy of Sciences*, 95(19):11158–11162, 1998.

[17] Andrea Thorn and George M. Sheldrick. Extending molecular-replacement solutions with SHELXE. *Acta Crystallographica Section D*, 69(11):2251–2256, Nov 2013.

[18] Martyn D Winn, Charles C Ballard, Kevin D Cowtan, Eleanor J Dodson, Paul Emsley, Phil R Evans, Ronan M Keegan, Eugene B Krissinel, Andrew GW Leslie, Airlie McCoy, et al. Overview of the CCP4 suite and current developments. *Acta Crystallographica Section D: Biological Crystallography*, 67(4):235–242, 2011.

[19] Jing Tao Yao, A.V. Vasilakos, and W. Pedrycz. Granular computing: Perspectives and challenges. *Cybernetics, IEEE Transactions on*, 43(6):1977–1989, Dec 2013.

[20] Adam Zemla. LGA: a method for finding 3D similarities in protein structures. *Nucleic Acids Research*, 31(13):3370–3374, 2003.

[21] Yang Zhang. Protein structure prediction: when is it useful? *Current opinion in structural biology*, 19(2):145–155, 2009.

[22] Yang Zhang. Interplay of I-TASSER and QUARK for template-based and ab initio protein structure prediction in CASP10. *Proteins: Structure, Function, and Bioinformatics*, 82:175–187, 2014.

[23] Yang Zhang and Jeffrey Skolnick. SPICKER: A clustering approach to identify near-native protein folds. *Journal of computational chemistry*, 25(6):865–871, 2004.